**AFRL-IF-RS-TR-2002-100**
**Final Technical Report**
**May 2002**

# TRUST MANAGEMENT IN OPEN SYSTEMS (TMOS)

**North Carolina State University**

**Sponsored by**
**Defense Advanced Research Projects Agency**
**DARPA Order No. 9804**

*APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.*

**AIR FORCE RESEARCH LABORATORY**
**INFORMATION DIRECTORATE**
**ROME RESEARCH SITE**
**ROME, NEW YORK**

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2002-100 has been reviewed and is approved for publication.

APPROVED:

Melvin J. Oster

**MELVIN J. OSTER**
**Project Engineer**

FOR THE DIRECTOR:

WARREN H. DEBANY, Jr., Technical Advisor
Information Grid Division
Information Directorate

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 074-0188*

| | | |
|---|---|---|
| **1. AGENCY USE ONLY (Leave blank)** | **2. REPORT DATE**<br>May 02 | **3. REPORT TYPE AND DATES COVERED**<br>Final Jun 98 - Dec 00 |

**4. TITLE AND SUBTITLE**
TRUST MANAGEMENT IN OPEN SYSTEMS (TMOS)

**5. FUNDING NUMBERS**
C - F30602-98-C-0222
PE - 33401G
PR - 9804
TA - TM
WU - OS

**6. AUTHOR(S)**

Vicki E. Jones, William H. Winsborough and Kent Seamons

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
North Carolina State University
Campus Box 7514
1 Leazar Hall
Raleigh, NC 27695-7514

**8. PERFORMING ORGANIZATION REPORT NUMBER**

**9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)**

Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington, VA 22203-1714

AFRL/IFGA
525 Brooks Road
Rome NY 13441-4505

**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

AFRL-IF-RS-TR-2002-100

**11. SUPPLEMENTARY NOTES**
AFRL Project Engineer: Melvin J. Oster, IFGA, 315-330-1870, osterm@rl.af.mil

**12a. DISTRIBUTION / AVAILABILITY STATEMENT**
Authorized for public release; distribution unlimited.

**12b. DISTRIBUTION CODE**

***13. ABSTRACT* (Maximum 200 Words)**

Distributed software subjects face the problem of determining one another's trustworthiness. The problem considered in the Trust Management in Open Systems (TMOS) project is management of the exchange of sensitive credentials between strangers for the purpose of property-based authentication and authorization. We designed a framework in which client and server establish mutual trust by exchanging credentials that are themselves protected resources. Protected resources are governed by role-based access control policies where roles are derived directly from property-based credentials. Within the framework, credentials are disclosed only to entities that meet the governing access control policies. By performing a sequence of credential exchanges, the framework establishes trust incrementally, enabling sensitive credentials to flow as required to meet the trust requirements of a desired transaction. In addition to a trust negotiation framework, we developed the concept of a negotiation strategy. A negotiation strategy controls the exchange of credentials. For instance, it determines how success and failure are detected. It also determines whether the exchange is guided by an exchange of explicit credential requests and, if so, the content of those requests. We formally specified and analyzed three negotiation strategies (the eager strategy, the parsimonious strategy, and the prudent strategy) and investigated hybrids of these strategies.

**14. SUBJECT TERMS**
Credentials, Management, Negotiation, Authorization, Authentication, Strategies

**15. NUMBER OF PAGES**
20

**16. PRICE CODE**

| | | | |
|---|---|---|---|
| **17. SECURITY CLASSIFICATION OF REPORT**<br>UNCLASSIFIED | **18. SECURITY CLASSIFICATION OF THIS PAGE**<br>UNCLASSIFIED | **19. SECURITY CLASSIFICATION OF ABSTRACT**<br>UNCLASSIFIED | **20. LIMITATION OF ABSTRACT**<br>UL |

# Abstract

*Distributed software subjects face the problem of determining one another's trustworthiness. The problem considered in the Trust Management in Open Systems (TMOS) project is management of the exchange of sensitive credentials between strangers for the purpose of property-based authentication and authorization. We designed a framework in which client and server establish mutual trust by exchanging credentials that are themselves protected resources. Protected resources are governed by role-based access control policies where roles are derived directly from property-based credentials. Within the framework, credentials are disclosed only to entities that meet the governing access control policies. By performing a sequence of credential exchanges, the framework establishes trust incrementally, enabling sensitive credentials to flow as required to meet the trust requirements of a desired transaction.*

*In addition to a trust negotiation framework, we developed the concept of a negotiation strategy. A negotiation strategy controls the exchange of credentials. For instance, it determines how success and failure are detected. It also determines whether the exchange is guided by an exchange of explicit credential requests and, if so, the content of those requests. We formally specified and analyzed three negotiation strategies---the eager strategy, the parsimonious strategy, and the prudent strategy---and investigated hybrids of these strategies.*

*We constructed two prototype systems that demonstrate trust negotiation and implement the eager negotiation strategy. The first system uses the trust policy language, developed at IBM Haifa Research Lab, to specify mappings from credentials to roles, and the trust establishment system, also from Haifa, to evaluate role membership questions. We deployed the IBM system in a scenario that illustrates a potential application of trust negotiation in a real-world situation. The second system, developed at North Carolina State University (NCSU), also implements the eager negotiation strategy but uses freely available components. The NCSU system uses an Apache web server and Java application programs.*

*Limited TMOS resources were used to investigate an authorization model for a public key management service. The model supports public key registration, lookup and revocation and private key escrow, protected use, and recovery. In the access control model proposed, policy is based on principal, ownership, and authority relationships on keys.*

# Table of Contents

1. Summary ........ 1
2. Introduction ........ 1
3. Credential-based Trust ........ 2
   3.1. When Credentials are Sensitive ........ 3
   3.2. Negotiation Architecture and Model ........ 4
      Trust Negotiation Model ........ 5
   3.3. Negotiation Strategy ........ 6
      An Eager Strategy ........ 6
      A Parsimonious Strategy ........ 7
      A Prudent Strategy ........ 10
      Hybrid Strategies ........ 10
   3.4. Credential Expression Languages ........ 10
   3.5. Demonstration Prototypes ........ 10
4. Public Key Management ........ 11
5. Conclusions ........ 11
6. References ........ 12

# List of Figures

Figure 1. Two credentials forming a chain. .......................................................................................... 3
Figure 2. The role of security agents in trust negotiation. ............................................................................ 4

## 1. Summary

We have designed a framework in which client and server establish mutual trust by exchanging credentials that are themselves protected resources. Protected resources are governed by role-based access control policies where roles are derived directly from property-based credentials. Within the framework, credentials are disclosed only to entities that meet the governing access control policies. By performing a sequence of credential exchanges, the framework establishes trust incrementally, enabling sensitive credentials to flow as required to meet the trust requirements of a desired transaction.

In addition to a trust negotiation framework, we have developed the concept of a negotiation strategy. A negotiation strategy controls the exchange of credentials. For instance, it determines how success and failure are detected. It also determines whether the exchange is guided by an exchange of explicit credential requests and, if so, the content of those requests. We formally specified and analyzed three negotiation strategies---the eager strategy, the parsimonious strategy, and the prudent strategy---and investigated hybrids of these strategies.

We have constructed two prototype systems that demonstrate trust negotiation and implement the eager negotiation strategy. The first system uses the trust policy language, developed at IBM Haifa Research Lab, to specify mappings from credentials to roles, and the trust establishment system, also from Haifa, to evaluate role membership questions. We have deployed the IBM system in a scenario that illustrates a potential application of trust negotiation in a real-world situation. The second system, developed at North Carolina State University (NCSU), also implements the eager negotiation strategy but uses freely available components. The NCSU system uses an Apache web server and Java application programs.

Limited TMOS resources were used to investigate an authorization model for a public key management service. The model supports public key registration, lookup and revocation and private key escrow, protected use, and recovery. In the access control model proposed, policy is based on principal, ownership, and authority relationships on keys.

## 2. Introduction

The primary contribution of the TMOS project has been in the area of automated trust negotiation. Limited resources were applied to public key management. Thus, this report focuses on trust negotiation research results.

Distributed software subjects face the problem of determining one another's trustworthiness. Current mainstream approaches to establishing trust presume that communicating subjects are already familiar with one another. There are essentially two approaches based on this assumption. The first is identity-based: identifying a subject is often a sufficient basis for doing business. The second is capability-based: subjects obtain capabilities that are specific to the resources they wish to use. Both approaches require that familiarity be established out of band. Identity- and capability-based approaches are both unable to establish trust between complete strangers. Other solutions are needed in open systems, such as the web, where the assumption of familiarity is invalid.

Property-based *digital credentials* [1] (or simply *credentials*) are the on-line analogues of paper credentials that people carry in their wallets. They present a promising approach to trust establishment in open systems. Credentials, which generalize the notion of attribute certificates [26], can authenticate not just the subject's identity, but arbitrary properties of a subject and its relationships with other subjects. Those properties can then be used, for instance, when a client attaches appropriate credentials to a service request, to support service authorization.

Trust establishment between strangers is particularly important in the context of e-business. Credential exchange between strangers promises to enable software agents to establish trust automatically with potential business partners. For instance, a software agent might be charged with finding new candidate suppliers of commodity goods and services. Even when an automatically generated list of such candidates eventually would be culled by a human, information such as requirements and availability of desired goods might be sensitive, requiring trust establishment as part of the automated process of identifying candidates.

The TMOS project investigated automated trust establishment between strangers through credential exchange when credentials are themselves potentially sensitive. A *sensitive credential* contains private information. For instance, access to a credential containing medical information could be restricted to primary care physicians and HMO staff. Access to a credit card credential could be limited to businesses authorized to accept a VISA card and that adhere to guidelines for securing credit card numbers. Prior trust establishment systems based on credential exchange have addressed credential sensitivity only manually, requiring a user at the client to decide which credentials to submit to each new service. Not only does this approach require human intervention. It provides the human no assistance in evaluating the trustworthiness of the server.

This report outlines an architecture for client-server applications in which client and server each establishes a *credential access policy* (CAP) for each of its credentials. A credential is disclosed only when its CAP is satisfied by credentials obtained from the opposing software agent. When an agent needs additional credentials, it can request them. Credentials flow between the client and server through a sequence of alternating credential requests and disclosures, which we call a *trust negotiation*. A formal, abstract model of trust negotiation is briefly discussed and then used to specify negotiation strategies.

A negotiation strategy determines characteristics of a negotiation such as which credentials are requested and disclosed, and when the negotiation is halted. This report informally discusses three negotiation strategies, finding them efficient and effective in establishing trust whenever possible.

Section 3 introduces credentials and explains how they can be used to establish trust between strangers. Section 3.1 introduces credential sensitivity and the problems it creates. Section 3.2 presents a trust negotiation architecture and an abstract model of trust negotiation that is used in Section 3.3, where eager, parsimonious, and prudent negotiation strategies are specified. Section 3.4 briefly discusses credential expression languages; Section 3.5 summarizes two demonstration prototypes; and Section 4 summarizes TMOS research in the area of public key management services.

## 3. Credential-based Trust

A credential is a digitally signed assertion by the *credential issuer* about the *credential owner*. Credentials can be made unforgeable and verifiable by using modern encryption technology: a credential is signed using the issuer's private key and verified using the issuer's public key [15]. A credential aggregates one or more *attributes* of the owner, each consisting of an attribute name/value pair and representing some property of the owner asserted by the issuer. For our purposes, a credential is specifically not required to identify the owner. Each credential also contains the public key of the credential owner. The owner can use the corresponding private key to answer challenges or otherwise demonstrate possession of that private key to establish ownership of the credential. The owner can also use the private key to sign another credential, owned by a third subject. In this way, *credential chains* can be created, with the owner of one credential being the issuer of the next credential in the chain.

Credential chains can be submitted to trace a web of trust from a known subject, the issuer of the first credential in the chain (e.g., subject A in Figure 1), to the *submitting subject*, in which trust is needed. The submitting subject is the owner of the last credential in the chain (e.g., subject C) and can demonstrate ownership of that credential, as outlined above. *Supporting* credentials are owned by subjects with whom the submitting subject has a direct or indirect relationship, and, although they are not owned by the submitting subject, the submitting entity does collect, keep, and submit copies of them. Each supporting credential contains the public key whose private-key mate signed the next credential in the chain, enabling reliable verification that the attribute claims made in that next credential were made by the owner of the supporting credential.

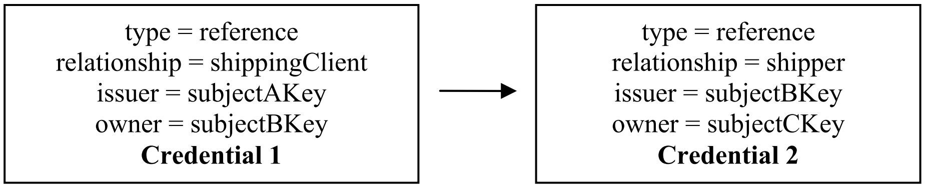

Figure 1. Two credentials forming a chain.
Credential 2 was issued by subject B, the owner of Credential 1. In Credential 1, subject A asserts that subject B is a consumer of shipping services. In Credential 2, subject B asserts that subject C is a shipper. If we trust subject A's judgment that subject B is a consumer of shipping, presumably subject B is in a position to know that subject C is a shipper. Additional credentials owned by subject B can be used to engender trust that subject B is a reliable authority on the asserted attributes of subject C.

The submitted credentials attempt to demonstrate a (possibly indirect) relationship between the submitting subject and the known subject that issued the first credential in the chain. The nature of that relationship can be inferred by inspecting the attributes of the credentials in the chain. Multiple chains can be submitted to establish a higher degree of trust or to demonstrate additional properties of the submitting subject and its relationships with known subjects.

A *credential expression,* $\psi$, is a logical expression over credentials with constraints on their attributes. A credential expression serves to denote the combinations of credentials, *C*, that satisfy it. We call those combinations the *solutions* of the expression. For the purpose of trust negotiation, credential expressions can be used to convey requests for credentials between client and server. In this context, credential expressions denote chains of credentials that end with credentials owned by the submitting subject. A credential expression can also be used as a *policy* governing access to a resource. Access to the resource is granted to a subject when a solution is presented that consists of one or more chains ending in credentials owned by the subject. The resource is *unlocked* by the solution.

A policy is *mobile* if it is sent from one subject to another as part of automatic or semiautomatic trust establishment. Mobile policies are used in prior systems to express requirements a client must meet to obtain service. When insufficient credentials accompany a service request, the server returns the *service-governing policy* (SGP). Communicated in this way, the SGP acts as a request for the credentials needed to unlock the resource. Such mobile policies enable clients to select a set of credentials whose submission will authorize the desired service. The client can then issue a second request for service with those credentials attached, and upon verifying the credentials, the server provides the desired service. Policy mobility has two significant advantages. First, it offloads from the server to the client the work of searching the client's credentials. Second, it enables trust to be established in the client without the client revealing irrelevant credentials.

### 3.1. When Credentials are Sensitive

A client wishing to do business with a new service may be unwilling to disclose sensitive credentials until some degree of trust has been established in that service. Current credential systems do not address credential sensitivity. The decision to disclose a sensitive credential to a new service is left up to a user at the client. More specifically, client-credential submission policies specify which credentials can be submitted with any request to a specified class of service and which credentials require explicit authorization before they are submitted. This mechanism requires a user be available to make trust decisions when new service classes are contacted. It does not address how the user decides to trust a service. The TMOS project developed a method of automating the establishment of trust between strangers through *incremental* exchange of sensitive credentials.

### 3.2. Negotiation Architecture and Model

A credential is protected by a CAP that controls the credential's disclosure based on credentials presented by the other negotiation participant. Throughout, credentials are disclosed only in observance of these CAPs.

Each negotiation participant is represented in trust negotiations by a *security agent* (SA), as in the simple negotiations of Ching et al. [5] and Winslett et al. [21]. The role of the SA is illustrated in Figure 2, which depicts the client security agent (101) and the server security agent (201) and several resources and contextual factors that each SA considers during negotiation. The client SA manages the disclosure of client credentials (102) and the server SA manages disclosure of server credentials (202). Like any protected resource, each credential is governed by an access policy (103, 203) that has the same form as a SGP. The CAP identifies credentials from the other negotiation participant that would unlock disclosure of the local credential to that subject.

The client (10) initiates the trust negotiation by making a service request. The client SA intercepts the request and relays it to the server SA. The application server (20) is accessible only via the server SA. Upon receiving a request for service (305), the server SA makes an authorization decision based on the appropriate SGP (206). When the client SA is familiar with the SGP, it can attach appropriate credentials (304) to the service request so that the service will be authorized. The server SA determines whether the credentials that arrive with the service request satisfy the SGP. If the policy is satisfied, the trust negotiation has completed successfully; the service is authorized and the request is forwarded to the application server, which provides the service to the client (300).

Initially, the client is unfamiliar with the SGP, so attaching satisfactory credentials to an initial service request is impractical. A trust negotiation strategy can overcome this problem by using mobile policies. When a server SA receives a request for service without sufficient credentials attached to satisfy the SGP, it sends the SGP to the client SA as a request for client credentials (302). The client SA can then select a combination of credentials that satisfies the SGP, and can attach those credentials (304) to a repetition of the original service request (305).

An important issue in this scenario not addressed in previous trust systems is how to enable the client SA to make independent trust decisions about which credentials to provide to an unfamiliar server. Our SAs use CAPs (103, 203) when selecting credentials to disclose. If the client SA cannot satisfy the SGP by using credentials whose CAPs are unprotected, it can, as the *negotiation instigator*, introduce further stages to the trust negotiation by requesting server credentials (303). These stages seek to build mutual trust through credential exchange, eventually to unlock client credentials that satisfy the SGP. Client

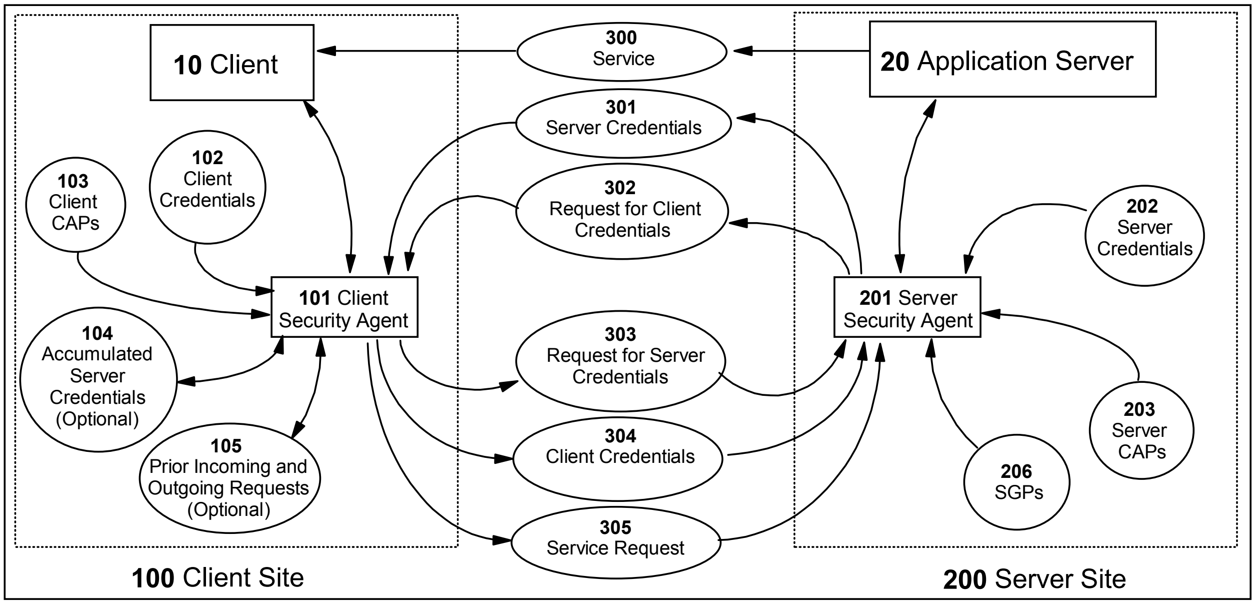

Figure 2. The role of security agents in trust negotiation.

credentials are unlocked by incoming server credentials (301); however, as an optimization, the client may also cache and use for this purpose server credentials it received in prior stages (104). (The abstract model of trust negotiation introduced in Section 0 does not capture this optimization.)

In each negotiation stage, the active subject responds to an incoming request for credentials either by providing credentials, by formulating a counter request for credentials (302, 303), or both. In some strategies, the client SA can also repeat one of its previous request (105) for credentials that has not yet been satisfied. By exchanging credentials and requests for credentials, the two SAs endeavor to establish trust required to authorize service. Eventually, either the negotiation succeeds or the client SA must abandon the attempt. The negotiation succeeds when the client SA satisfies the SGP by disclosing sufficient unlocked credentials. At the same time, the client SA repeats the original service request (305), this time with sufficient credentials attached (304) to authorize service.

## Trust Negotiation Model

Critical for analysis of potential negotiation strategies is the trust negotiation model. The basic model introduced by the TMOS project is discussed below. The abstract model formalizes a trust negotiation as a sequence of credential disclosures that alternate between the two participants, optionally augmented by a sequence of credential requests that serve to guide the disclosures.

The participants in a trust negotiation are the client and server. Each owns a finite set of credentials, which we denote by ClientCreds and ServerCreds, respectively. Access to each credential $c$ in ClientCreds or ServerCreds is governed by a policy, denoted $\mathsf{gov}_{\mathsf{client}}(c)$ or $\mathsf{gov}_{\mathsf{server}}(c)$, respectively. If a credential expression, $\psi$, is satisfied by a set of credentials $C$, we write $\mathbf{sat}(C, \psi)$. Credential expressions are required to be *monotonic*; that is, if $C \subseteq C'$, then $\mathbf{sat}(C, \psi)$ implies $\mathbf{sat}(C', \psi)$. We write $\psi \equiv \psi'$ if for all credential sets $C$, $\mathbf{sat}(C, \psi)$ *iff* $\mathbf{sat}(C, \psi')$. We do not specify a language of credential expressions here, though an example language is outlined in Section 3.4.

If Creds is any finite set of credentials and $\mathbf{C}$ is any set of subsets of Creds, then there exists $\psi$ such that for all $C \subseteq$ Creds, $\mathbf{sat}(C, \psi)$ *iff* $C \in \mathbf{C}$.

The purpose of this requirement is to ensure that arbitrary sets of credentials can be specified in negotiation strategies where credential expressions are transmitted for the purpose of requesting credentials from other security agents.

We write $\mathbf{unlocked}(c, C)$ either if $C \subseteq$ ClientCreds, $c \in$ServerCreds, and $\mathbf{sat}(C, \mathsf{gov}_{\mathsf{server}}(c))$, or if $C \subseteq$ ServerCreds, $c \in$ClientCreds, and $\mathbf{sat}(C, \mathsf{gov}_{\mathsf{client}}(c))$. If $\mathbf{unlocked}(c, \varnothing)$, $c$ is *unprotected*. Extending this relation to sets of credentials, $C'$, we write $\mathbf{unlocked}(C', C)$ if $\mathbf{unlocked}(c, C)$ holds for each $c \in C'$.

A *trust negotiation* is given by a sequence of credential disclosures, $\{C_i\}_{i\in[0,m]} = C_0,\ldots,C_m$, for some natural number $m$. (Throughout we use the notation $[i, j]$ to denote the integer interval from $i$ to $j$, inclusive.) Each disclosure models a message containing credentials. $C_0$ models a disclosure by the client to the server. The disclosures then alternate between the two subjects, which we formalize as follows. Define $\{\mathsf{AltCreds}_i\}_{0\le i}$ by:

$$\mathsf{AltCreds}_i = \begin{cases} \mathsf{ClientCreds}, \text{if } i \text{ is even} \\ \mathsf{ServerCreds}, \text{if } i \text{ is odd} \end{cases}$$

We require $C_i \subseteq \mathsf{AltCreds}_i$ for $0 \le i \le m$.

The credentials in each disclosure are required to be unlocked by credentials from the other negotiation participant in the previous disclosure, which means that the first disclosure consists entirely of credentials that are unprotected. That is, we have $\mathbf{unlocked}(C_0, \varnothing)$ and $\mathbf{unlocked}(C_{i+1}, C_i)$ for all $0 \le i < m$. Any disclosure can be empty, provided the subsequent disclosure consists of unprotected credentials.

Both client and server may set *trust requirements* that a trust negotiation may or may not succeed in establishing. Trust requirements are represented by credential expressions. The trust requirement of primary concern in this article is the server's policy governing access to a service: the SGP. However, a client might also set a trust requirement that it enforces before doing business with a server. In negotiation strategies where trust requirements provide a goal that focuses the credential exchange, we call the trust requirements *trust targets*.

A trust negotiation *satisfies a server-set trust requirement*, $\psi$, if some client disclosure satisfies $\psi$, *i.e.,* if $\mathsf{sat}(C_j, \psi)$ for some even $j \in [0, m]$. A trust negotiation *satisfies a client-set trust requirement*, $\psi$, if some server disclosure satisfies $\psi$, *i.e.*, if $\mathsf{sat}(C_j, \psi)$ for some odd $j \in [0, m]$. In either case, we say that $\{C_i\}_{i\in[0,m]}$ *satisfies* $\psi$.

In one of the trust negotiation strategies introduced in Section 0, disclosures are guided by *credential requests* that are also exchanged by the negotiation participants. Credential requests are formalized by a sequence of credential expressions that accompanies the trust negotiation and that has the same length as the sequence of disclosures. For a given trust negotiation, $\{C_i\}_{i\in[0,m]}$, an accompanying sequence of credential requests has the form $\{\psi_i\}_{i\in[0,m]}$.

### 3.3. Negotiation Strategy

In our trust negotiation architecture, the negotiation strategy determines the search for a successful negotiation. The strategy determines which credentials are disclosed, when they are disclosed, and which credentials are requested from the other subject to unlock local credentials. Successful trust negotiation is not always possible. One subject or the other may not possess needed credentials, or subjects may govern their credentials by policies that, together, impose cyclic dependencies. The strategy determines when the negotiation instigator—the client in our architecture—gives up on a negotiation.

Some desirable properties of negotiation strategies are as follows. A strategy should lead to a successful negotiation whenever one exists; that is, it should be *complete*. It should terminate with failure when success is impossible. Ideally, it should enforce a need-to-know policy, avoiding disclosure of credentials that are not needed for the negotiation to succeed and disclosing no credentials when the negotiation fails. Finally, a strategy should be efficient, giving a reasonable bound on the number of messages that must flow during the negotiation. We analyze the extent to which these properties are satisfied by using the abstract model defined in Section 3.2.

Within the context of the abstract model, we identify each negotiation strategy with a set of trust negotiations. This high level of abstraction focuses our attention on the essential relationships between CAPs and the disclosures and requests that flow between client and server SAs in each strategy. We defined and analyzed three negotiation strategies, discussed briefly below. A complete discussion of these strategies is not included in this report but can be found in [18] and [23].

## An Eager Strategy

In the eager strategy, two security agents take turns sending every credential they have that is currently unlocked. As credentials are exchanged, more credentials become unlocked. The client terminates a negotiation when it receives a set of credentials from the server that it has already received (no new credentials) or the set it receives unlocks no new client credentials. This strategy, as formalized here, does not focus on a particular trust target, but simply expedites a maximal credential exchange.

**Definition (*Eager Negotiation*):** A trust negotiation, $\{C_i\}_{i\in[0,m]}$, is an *eager negotiation* if,

1. $C_0$ is the **maximal** set such that $\mathsf{unlocked}(C_0, \varnothing)$ and, for all $i \in [1, m]$, $C_i$ is the **maximal** set such that $\mathsf{unlocked}(C_i, C_{i-1})$, and,
2. for all $i \in [1, m-3]$, $C_i \, \gamma \, C_{i+2}$, and,
3. if $m$ is even, $C_{m-2} \, \gamma \, C_m$.

Since it is the client that detects termination, the last disclosure from the server may repeat the server's prior disclosure. In the following discussion, ClientCreds, ServerCreds, $\text{gov}_{\text{client}}$, and $\text{gov}_{\text{server}}$ are fixed but arbitrary.

In [18] three theorems related to Eager strategy properties are presented and analyzed. Here we omit the proofs.

- **Theorem 1 (*Efficiency of eager negotiation*):** The length, $m$+1, of any eager negotiation, $\{C_i\}_{i\in[0,m]}$, is at most 2×**min**(|ClientCreds|+1, |ServerCreds|+1).
- **Theorem 2 (*Uniqueness of eager negotiation*):** There is a unique maximal length eager negotiation.
- **Theorem 3 (*Completeness of eager negotiation*):** For any credential expression if there exists any trust negotiation satisfying the expression, then the maximal length eager negotiation satisfies the expression.
- **Corollory 4 (*Minimality of length of eager negotiation*):** For any credential expression, $\psi$, if there exists any trust negotiation satisfying $\psi$, then there exists an eager negotiation of equal or lessor length satisfying $\psi$.

The strengths of the eager strategy are its simplicity and the fact that no information about credentials possessed is disclosed unless the CAP of the credential in question is satisfied. Its weakness is that it discloses credentials without regard to their relevance to the present negotiation: there is no provision for disclosing on a need-to-know basis.

## A Parsimonious Strategy

Eager negotiations begin exchanging credentials essentially immediately. They make little or no use of credential requests; although one of the variants presented above does call for the SGP to flow as a credential request, even there, until the SGP can be satisfied by unlocked client credentials, all unlocked credentials are exchanged without regard for any credential request. The *parsimonious strategy* differs from the eager strategy in these respects. An intuitive explanation is presented here. The formal definition can be found in [18].

1. Requests are exchanged to guide the negotiation toward satisfying a particular trust target. In general, this trust target could be a SGP or a trust requirement set by the client. To simplify the presentation, we assume the trust target is a SGP. Under this assumption, the first credential request from the server is the trust target (i.e., the SGP).

2. When and if a request is sent that can be satisfied by unprotected (and therefore unlocked) credentials, the negotiation reaches the *point of confidence.* Corollary 7 and Theorem 8 below together show that when this occurs, the negotiation is bound to succeed.

3. Initial credential disclosures are empty in each stage up to and including the point of confidence. If the point of confidence is never reached, the negotiation terminates without disclosing any credentials.

4. Prior to the point of confidence, each successive credential request is derived from its predecessor in a manner that makes satisfying that request a necessary and sufficient condition for a disclosure to unlock credentials that satisfy the predecessor.

5. After the point of confidence is reached, the client resends its prior requests, going through them backwards, at the same time disclosing appropriate credentials to unlock solutions to those requests.

6. As mentioned above in point 2, when and if a request is sent that can be satisfied by a set of unprotected credentials, a **minimal** such set is disclosed in the next stage. Each successive step also discloses a minimal credential set that satisfies a credential request, working backwards through the requests that were issued prior to reaching the point of confidence. The client, which drives the negotiation, will have recorded each of the requests that has flowed. It refers to requests it received from the server when selecting its own credential disclosures; it resends the requests it sent to the server, as outlined in point 5 above. Each disclosure unlocks the next, until a disclosure satisfying the original trust target is unlocked.

**Definition (*Parsimonious Negotiation*):** Let $\psi$ be a credential expression and $\{C_i\}_{i\in[0,m]}$ be a trust negotiation. $\{C_i\}_{i\in[0,m]}$ is a *parsimonious negotiation with respect to the trust target*, $\psi$, if it is accompanied by a sequence of credential requests, $\{\psi_i\}_{i\in[0,m]}$, and the following six requirements are satisfied:

1. $\psi = \psi_1$.
2. If there exists a $j \in [1, m]$ and a $C \subseteq \mathsf{AltCreds}_{j+1}$ such that $\mathbf{sat}(C, \psi_j)$ and $\mathbf{unlocked}(C, \varnothing)$, then, letting $k$ be the least such $j$, we say that the negotiation *reaches the point of confidence at stage k*. Otherwise, we let $k = m$.
3. For all $i$, $1 \le i \le k$, $C_i = \varnothing$.
4. For all $i$, $1 \le i < k$, $\psi_i$ and $\psi_{i+1}$ have the following relationship:
5. For all $C \subseteq \mathsf{AltCreds}_{i+2}$, we have $\mathbf{sat}(C, \psi_{i+1})$ *iff* there exists a $C' \subseteq \mathsf{AltCreds}_{i+1}$, such that $\mathbf{sat}(C', \psi_i)$ and $\mathbf{unlocked}(C', C)$.
6. After reaching the point of confidence, prior client requests (which have even indices) are replayed. If $k$ is even, we require $\psi_{k+j} = \psi_{k-j}$ for even $j$, $2 \le j < m-k$ (if any). If $k$ is odd, we require $\psi_{k+j} = \psi_{k-j}$ for odd $j$, $1 \le j < m-k$ (if any).
7. If the negotiation reaches the point of confidence at stage $k$, we require that for all $j$, $0 \le j < m-k$, (if any) $C_{k+j+1}$ is a **minimal** (under $\subseteq$) subset of $\mathsf{AltCreds}_{k+j+1}$ satisfying $\mathbf{sat}(C_{k+j+1}, \psi_{k-j})$ (and, as for any trust negotiation, $\mathbf{unlocked}(C_{k+j+1}, C_{k+j})$ (recall that $C_k = \varnothing$)).

In [18] five theorems with associated corollaries and definitions related to parsimonious strategy properties are presented and analyzed. Here we omit the proofs.

In Requirement 4, for any $\psi_i$, we know that $\psi_{i+1}$ exists by the expressivity requirement on credential expression languages (see Section 0). Since each parsimonious negotiation, $\{C_i\}_{i\in[0,m]}$, has an associated sequence of credential requests, $\{\psi_i\}_{i\in[0,m]}$, we often refer to the parsimonious negotiation as the pair $\langle\{C_i\}_{i\in[0,m]}, \{\psi_i\}_{i\in[0,m]}\rangle$. In the following discussion, ClientCreds, ServerCreds, $\mathsf{gov}_{\mathsf{client}}$, and $\mathsf{gov}_{\mathsf{server}}$ are fixed but arbitrary.

- **Theorem 5 (*Determinacy of Requests in Parsimonious Negotiation*):** Given two parsimonious negotiations, $\langle\{C_i\}_{i\in[0,m]}, \{\psi_i\}_{i\in[0,m]}\rangle$ and $\langle\{C'_i\}_{i\in[0,n]}, \{\psi'_i\}_{i\in[0,n]}\rangle$, with equivalent trust targets, $\psi_1 \equiv \psi_1'$, we have $\psi_i \equiv \psi_i'$ for all $i$, $1 \le i \le k$, where $k$ is either the stage where one of the negotiations reaches the point of confidence, or $m$, or $n$, whichever is least.

- **Theorem 6 (*When success is possible, parsimonious negotiations efficiently reach the point of confidence*):** Given any credential expression $\psi$, if there exists a trust negotiation that satisfies $\psi$, then we have the following:
    - There exists a natural number $k \le 2\times\mathbf{min}(|\mathsf{ClientCreds}|+1, |\mathsf{ServerCreds}|+1)$ such that every parsimonious negotiation $\langle \{C_i\}_{i\in[0,m]}, \{\psi_i\}_{i\in[0,m]}\rangle$, with $k \le m$ and trust target $\psi$, reaches the point of confidence at stage $k$; and
    - Every parsimonious negotiation $\langle \{C_i\}_{i\in[0,m]}, \{\psi_i\}_{i\in[0,m]}\rangle$ with trust target $\psi$ that has not reached the point of confidence (*i.e.*, $m < k$) can be extended to a parsimonious negotiation that reaches the point of confidence at stage $k$.
- **Definition (Stuck Parsimonious Negotiation):** Let $\langle \{C_i\}_{i\in[0,m]}, \{\psi_i\}_{i\in[0,m]}\rangle$ be a parsimonious negotiation with respect to some trust target $\psi$. $\langle \{C_i\}_{i\in[0,m]}, \{\psi_i\}_{i\in[0,m]}\rangle$ is *stuck* if it reaches the point of confidence at some stage $k$, $m < 2k$, and there is no $C' \subseteq \mathsf{AltCreds}_{m+1}$ with $\mathbf{sat}(C', \psi_{2k-m})$ and $\mathbf{unlocked}(C', C_m)$.
- **Theorem 7 (Parsimonious negotiations are not stuck):** No parsimonious negotiation is stuck.
- **Corollary 8 (Parsimonious negotiations that reach the point of confidence can be extended):** Every parsimonious negotiation that reaches the point of confidence at some stage $k$ can be extended to form a parsimonious negotiation of length $2k+1$.
- **Theorem 9 (Parsimonious negotiations that reach the point of confidence and are sufficiently long satisfy their trust targets):** Every parsimonious negotiation $\langle \{C_i\}_{i\in[0,m]}, \{\psi_i\}_{i\in[0,m]}\rangle$ that reaches the point of confidence at some stage $k$ and has $m = 2k$ satisfies its trust target.
- **Corollary 10 (Completeness and efficiency of parsimonious negotiation):** Given any credential expression $\psi$, if there exists a trust negotiation that satisfies $\psi$, then there exists a natural number $k \le 2\times\mathbf{min}(|\mathsf{ClientCreds}|+1, |\mathsf{ServerCreds}|+1)$ such that every parsimonious negotiation with trust target $\psi$ and length $2k+1$ satisfies $\psi$. Moreover, every parsimonious negotiation with trust target $\psi$ and length less than $2k+1$ can be extended to one with length $2k+1$.
- **Theorem 11 (Local minimality of disclosures in parsimonious negotiation):** In a parsimonious negotiation, no credential is disclosed unless and until the point of confidence is reached, at which time successful negotiation is guaranteed. Then, each disclosure consists of a minimal (under $\subseteq$) set of credentials sufficient to unlock either the next credential disclosure or the desired service.

Any deployment of the parsimonious strategy should take advantage of the fact that, if a successful negotiation exists, the initial exchange of credential requests will encounter a request that can be satisfied by unprotected credentials within the first $2\times(|\mathsf{ClientCreds}|+1)$ requests. If such a request has not occurred, the negotiation should be terminated.

## A Prudent Strategy

Eager and parsimonious negotiations may either fail when in fact success is possible, disclose irrelevant credentials, or have a high communication complexity. The *prudent negotiation strategy (PRUNES)* guarantees that trust is established, if allowed by the credential disclosure policies. PRUNES makes sure that no irrelevant credentials are disclosed during trust negotiations and is efficient: in the worst case, the communication complexity is $O(n^2)$ and the computational complexity is $O(nm)$, where $n$ is the number of credentials and $m$ is the size of the credential disclosure policies in disjunctive normal form.

Two theorems regarding PRUNES are defined below. A thorough discussion of the strategy and proofs of these theorems are available in [23].

- **Theorem 12:** The worst case communication complexity (total number of messages, total size of messages, and number of rounds) of PRUNES is $O(n^2)$, where $n$ is the total number of credentials requested during the negotiation.

- **Theorem 13:** The computational complexity of PRUNES is $O(nm)$, where $n$ is the total number of credentials and $m$ is the total size of the policies of both parties.

## Hybrid Strategies

Eager and parsimonious strategies can be combined in an effort to use each with the credentials for which it is better suited. CAPs can be made two-part, comprising not only a credential expression, but also a flag to select between parsimonious and eager disclosure. A credential flagged for eager disclosure would be disclosed freely to all sites that present credentials that satisfy the credential-expression component of its CAP. One flagged "parsimonious" would be disclosed only as part of a locally minimal exchange and successful negotiation. A hybrid strategy begins with a phase that uses an eager strategy to attempt to negotiate using only credentials flagged for eager disclosure. If success is possible using just those credentials, the negotiation succeeds during the eager phase. If not, phase two uses a parsimonious strategy to attempt to establish trust by using all credentials. Phase two takes advantage of credentials exchanged during phase one. In a hybrid negotiation, the client determines when phase one has completed unsuccessfully and phase two begins. The client indicates in each request to the server which strategy it is currently employing.

### 3.4. Credential Expression Languages

Our focus during the TMOS project was not on credential expression languages. The extensions we made to existing approaches [17] are presented in [18]. However, two language features, introduced and discussed in [18], reflect important issues that should guide further work in policy languages for trust negotiation. First, role attributes enhance expressiveness. Second, the monotonic relationship between credentials and access is essential in any context where a subject can withhold disclosure of its own credentials.

An *authorization policy* defined in our *Role-based Authorization Language* (*RAL*) consists of a *role-constraint expression*, which expresses requirements for access to the service or credential that it governs. These requirements are expressed in terms of roles of the subject seeking access. These roles are defined by an *authentication policy* written in our *Attribute-based Authentication Language* (*AAL*). An AAL authentication policy assigns a subject to roles based on subject properties derived from credentials owned by the subject and from the roles of the issuers of those credentials. This assignment is independent of the question of the subject's access to the service. Thus, an AAL role is not a capability, but represents a derived attribute of the subject.

### 3.5. Demonstration Prototypes

We constructed two prototype systems that demonstrate trust negotiation and implement the eager negotiation strategy. The first system uses the trust policy language, developed at IBM Haifa Research

Lab, to specify mappings from credentials to roles, and the trust establishment system, also from Haifa, to evaluate role membership questions. We have deployed the IBM system in a scenario that illustrates a potential application of trust negotiation in a real-world situation. The IBM system is described at a high level in [18]. The second system, developed at North Carolina State University (NCSU), also implements the eager negotiation strategy but uses freely available components. The NCSU system using, an Apache web server and Java application programs, is described in [12].

## 4. Public Key Management

The negotiation strategies described above depend on cryptographically secure public key services. Thus, limited TMOS resources were applied to the problem of public key management services. While the mechanics of certifying and revoking public keys, and escrowing and recovering private keys have been widely explored, less attention has been paid to access control frameworks for regulating access to stored keys by different parties. In [15] we proposed such a framework for a key management service supporting public key registration, lookup, and revocation, and private key escrow, protected use (e.g., to decrypt selected messages), and recovery. The access control model proposed uses policies based on principal, ownership, and authority relationships on keys. The model allows owners to grant to others (and revoke) privileges to execute different actions on their keys. The simple authorization language is very expressive, enabling the specification of authorizations for composite subjects that can be fully specified (ground) or partially specified, thus making the authorizations applicable to all subjects satisfying some conditions.

## 5. Conclusions

We developed a model and architecture for negotiating mutual trust between clients and servers through an incremental exchange of potentially sensitive credentials. We specified and analyzed three negotiation strategies, one eager and one parsimonious (*i.e*., stingy) with credential disclosures and one efficient and complete. The eager strategy negotiates efficiently, succeeding whenever possible. Its participants exchange no credential requests, nor otherwise attempt to minimize credential disclosures. The drawback is that many credentials are disclosed unnecessarily. However, the strategy reveals no information about any credential that a subject possesses until the credential's CAP is satisfied. There is an advantage in this: credential requests exchanged in the parsimonious strategy could in principle reveal a great deal about which credentials a subject has.

The parsimonious strategy conducts a minimal-length exchange in which each disclosure is a locally minimal set. It does this by conducting an exchange of credential requests that in effect considers every possible successful exchange. It remains open how to ensure that the union of each participant's disclosures is minimal—or whether this is even possible. Because agents using the parsimonious strategy exchange credential requests, even when negotiation fails, they provide one another with information as to the credentials that, if obtained, might enable successful negotiation in the future. The credentials that would enable future success could be new credentials issued to one of the agents, or they could be supporting credentials that document the properties of credential issuers.

The prudent strategy is complete in that trust is established if allowed by the credential disclosure policies. It also makes sure that no irrelevant credentials are disclosed during trust negotiations. Is efficient: in the worst case, the communication complexity is $O(n^2)$ and the computational complexity is $O(nm)$, where $n$ is the number of credentials and $m$ is the size of the credential disclosure policies in disjunctive normal form.

The strategies presented here assume that both participants cooperate in using the same strategy. Further research is required to determine whether and how that assumption can be relaxed or well justified. A parsimonious negotiation guarantees locally minimal credential disclosure only when both parties "bargain in good faith." This means that each SA assumes the following about the other SA. When the other SA responds to an incoming request by issuing a counter request, if that SA subsequently receives credentials that satisfy the counter request, together with a repetition of the original request, it will return credentials that satisfy that original request. One advantage of a hybrid negotiation strategy is that the eager phase could be used to establish trust that the other negotiation partner will bargain in good faith before entering a parsimonious negotiation.

This research focused on managing sensitive credentials. Further work is needed in the management of policy information. Negotiation strategies that exchange policy content introduce trust issues that have not yet been addressed. Policy owners may need to protect sensitive policy content. Security agents receiving mobile policy content may need to verify its authenticity and integrity.

## 6. References

[1] E. Bina, V. Jones, R. McCool, and M. Winslett, "Secure Access to Data Over the Internet," *Proceedings of the Third ACM/IEEE International Conference on Parallel and Distributed Information Systems*, Austin, Texas, Sept., 1994.

[2] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. Keromytis, "The KeyNote Trust Management System," work in progress, Internet Draft, March 1999.

[3] M. Blaze, J. Feigenbaum, and A. D. Keromytis, "KeyNote: Trust Management for Public-Key Infrastructures," Cambridge 1998 Security Protocols International Workshop, England, 1998.

[4] M. Blaze, J. Feigenbaum, and J. Lacy, "Decentralized Trust Management," *1996 IEEE Conference on Privacy and Security*, Oakland, 1996.

[5] N. Ching, V. Jones, and M. Winslett, "Authorization in the Digital Library: Secure Access to Services across Enterprise Boundaries," *Proceedings of ADL '96 --- Forum on Research and Technology Advances in Digital Libraries*, Washington, DC, May 1996. Available at http://drl.cs.uiuc.edu/security/pubs.html.

[6] T. Dierks, C. Allen, "The TLS Protocol Version 1.0," draft-ietf-tls-protocol-06.txt, Nov. 12, 1998.

[7] S. Farrell, "TLS Extensions for Attribute Certificate Based Authorization," draft-ietf-tls-attr-cert-01.txt, August 20, 1998.

[8] A. Frier, P. Karlton, and P. Kocher, "The SSL 3.0 Protocol," Netscape Communications Corp., Nov., 1996.

[9] W. Johnston, S. Mudumbai, and M. Thompson, "Authorization and Attribute Certificates for Widely Distributed Access Control," Proceedings of the IEEE 7th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises—WETICE '98.

[10] N. Li, J. Feigenbaum, and B. Grosof, "A Logic-based Knowledge Representation for Authorization with Delegation" (Extended Abstract), *Proceedings of the 12th Computer Security Foundations Workshop*, IEEE Computer Society Press, Los Alamitos, 1999, pp. 162-174. Full paper available as IBM Research Report RC21492(96966).

[11] N. Li, B. Grosof, and J. Feigenbaum, "A Practically Implementable and Tractable Delegation Logic," to appear in *Proceedings of the 2000 IEEE Symposium on Security and Privacy*.

[12] F. Lin, "Establishing Trust: Negotiating Disclosure of Sensitive Information," Master's Thesis, Department of Computer Science, North Carolina State University, May 2000.

[13] A. Herzberg, J. Mihaeli, Y. Mass, D. Naor, and Y. Ravid, "Access Control Meets Public Key Infrastructure, Or: Assigning Roles to Strangers," to appear in *2000 IEEE Symposium on Security and Privacy*, May 2000. Available at http://www.hrl.il.ibm.com/.

[14] "The Trust Policy Language," IBM Haifa Research Laboratory (http://www.hrl.il.ibm.com/), email contact: YOSIMASS@il.ibm.com.

[15] P. Samarati, M. Reiter, and S. Jajodia , "An Authorization Model for a Public Key Management Service," submitted to *ACM Transactions on Information and System Security*.

[16] B. Schneier, Applied Cryptography, John Wiley and Sons, Inc., second edition, 1996.

[17] K. Seamons, W. Winsborough, and M. Winslett, "Internet Credential Acceptance Policies," *Proceedings of the 2nd International Workshop on Logic Programming Tools for Internet Applications*, July, 1997. Available at http://clement.info.umoncton.ca/~lpnet/proceedings97/.

[18] W. Winsborough, K. Seamons, V. Jones, "Automated Trust Negotiation," ACM Transactions on Information and System Security, submitted April 2000.

[19] W. Winsborough, K. Seamons, and V. Jones, "Automated Trust Negotiation: Managing Disclosure of Sensitive Credentials," Transarc Research White Paper, May 1999.

[20] W. Winsborough, K. Seamons, and V. Jones, "Negotiating Disclosure of Sensitive Credentials," *Second Conference on Security in Communication Networks* (SCN '99), September, 1999.

[21] W. Winsborough, K. Seamons, and V. Jones, "Automated Trust Negotiation," *DARPA Information Survivability Conference and Exposition* (DISCEX '2000), January, 2000.

[22] M. Winslett, N. Ching, V. Jones, and I. Slepchin, "Using Digital Credentials on the World-Wide Web," *Journal of Computer Security*, **5**, 1997, 255-267.

[23] T. Yu, X. Ma, and M. Winslett, "PRUNES: An Efficient and Complete Strategy for Automated Trust Negotiation over the Internet," *Proceedings of the 2000 Computer and Communications Security Conference*, 2000.

[24] P. Zimmerman, PGP User's Guide, MIT Press, Cambridge, 1994.

[25] Simple Public Key Infrastructure (SPKI), http://www.ietf.org/html.charters/spki-charter.html.

[26] International Telecommunication Union, Recommendation X.509 - Information Technology - Open Systems Interconnection - The Directory: Authentication Framework, August 1997.

# List of Abbreviations

| | |
|---|---|
| AAL | attribute-based authentication language |
| CAP | credential access policy |
| NCSU | North Carolina State University |
| PRUNES | prudent negotiation stragegy |
| RAL | role-based authorization language |
| SA | security agent |
| SGP | service-governing policy |
| TMOS | Trust Management in Open Systems |